UNIVERSITY OF BIRMINGHAM

# Numerical Analysis

Dr. David Forest

**University of Birmingham**

MSc Physics and Technology of Nuclear Reactors

# Syllabus

Part 1 – Fitting and Errors

A. Least squares

B. Weighted least squares

C. Covariance

Part II – Numerical Methods

D. Finite differences

E. Interpolation

F. Differentiation by finite differences

G. Numerical integration

H. I. J. Numerical solution of differential equations

# Delivery and Assessment

Weekly practical sessions – 2 hrs

Worksheet with non-assessed exercises

Note: some worksheets are longer than others

Exercises to be completed using EXCEL

No programming experience necessary

Helps to visualise the problem

EXCEL has some useful built-in tools – e.g. matrix manipulation

Downside: hard to debug (ask a demonstrator)

Helpful to have attempted the worksheet before the session

Assessment

Selection of problems similar to some of the exercises

Set over the Christmas break to be handed-in in January

# Session A – Least Squares Fitting

## Outline of the problem

You have some data (set of x and y values)

You suspect that there is a relationship between them

For example, it could be linear: $y = mx + c$

Typically the values of the parameters m and c are of interest

However, the data is not perfect (e.g. due to measurement errors)

The points don't all lie on a straight line

Q. How do you find the line of 'best fit'?

A. Use the method of least-squares (this session)

Q. How do you cope if some points are less reliable than others?

A. Use the method of weighted-least-squares (session B)

Q. How does the error on y reflect on the values of m and c?

A. Through the error (covariance) matrix (session C)

# Fitting a straight line

## Nomenclature

You will be used to writing the equation of a straight line as:

$y = mx + c$

The straight line is a polynomial of order 1 ($x^n$; n=1)

We will generalise this so we can consider any order polynomial

$y = p_0 + p_1 x$

$y = p_0 + p_1 x + p_2 x^2$

… etc …

This will be useful when we introduce matrix methods

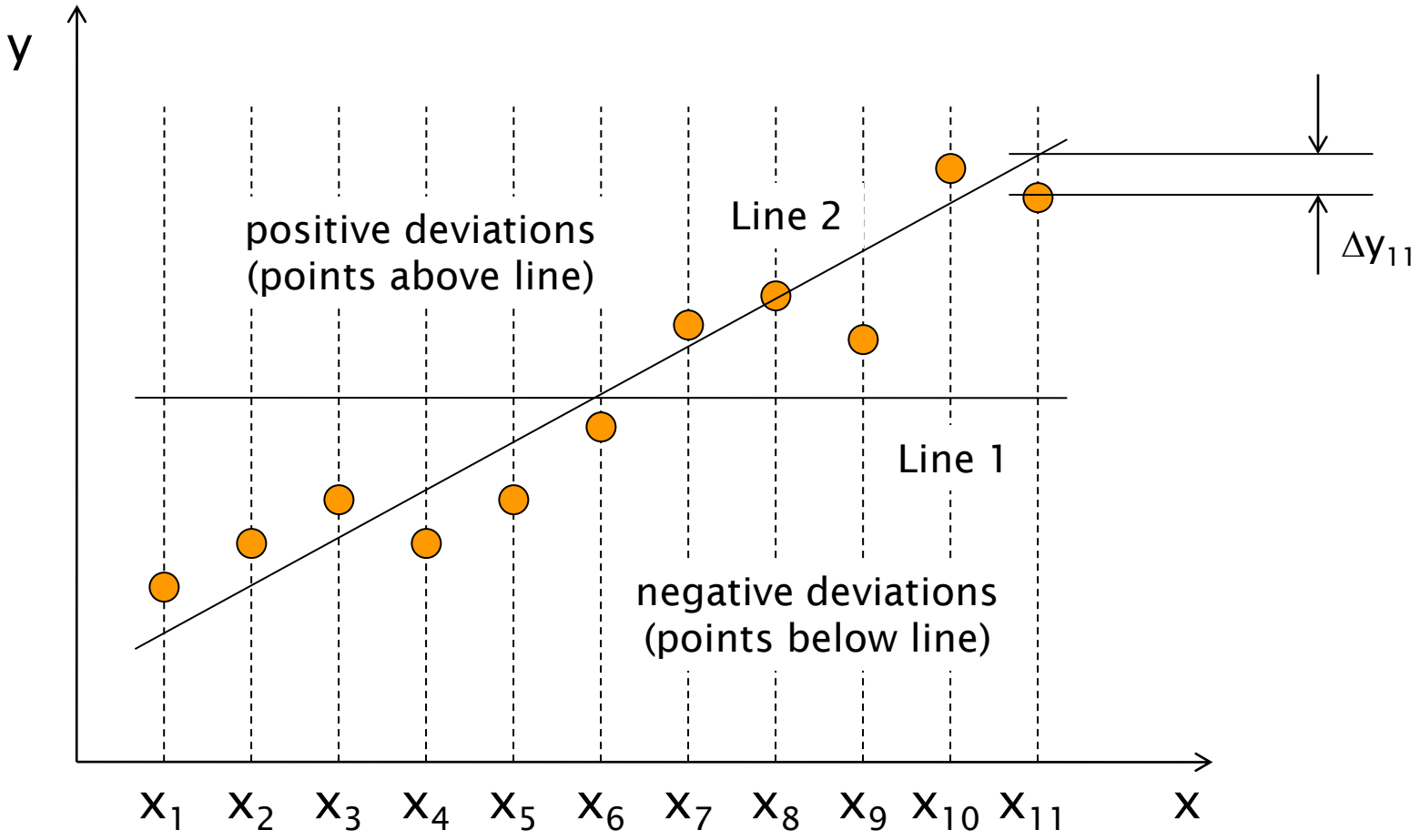
Next we need to define what we mean by 'best fit'

This is where the *least-squares* method comes in

# Method of least squares

## Definition of 'best fit'

Aim to minimise the *sum* of the *square* of the *deviations* ($\Delta y$) between each point and the line



# Method of least squares

## Let's write this mathematically

The expression for the sum of least square deviations is

$$S = \sum_{i=1}^{n} [y_i - y_{fit}]^2$$ Sum over all points

$$S = \sum_{i=1}^{n} [y_i - (p_1 x_i + p_0)]^2$$ For a straight line $y_{fit} = p_1 x + p_0$

We have two free parameters: the slope ($p_1$) and the intercept ($p_0$)

Finding the 'best fit' line involves minimising $S$ w.r.t. $p_1$ and $p_0$

That is, we want to find the values of $p_1$ and $p_0$ that satisfy

$$\frac{\partial S}{\partial p_1} = \frac{\partial S}{\partial p_0} = 0$$

Ok, let's do it … (see worksheet) …

# Method of least squares

## What do we end up with?

What we end up with is a pair of simultaneous equations:

$$p_1 \sum_{i=1}^{n} x_i^2 + p_0 \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i y_i$$

$$p_1 \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} p_0 = \sum_{i=1}^{n} y_i$$

Note: $\sum_{i=1}^{n} p_0 = n p_0$

Rearranging we can solve for $p_1$ and $p_0$

$$p_1 = \frac{\left[ n \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i \right]}{\left[ n \sum_{i=1}^{n} x_i^2 - \left( \sum_{i=1}^{n} x_i \right)^2 \right]}$$

$$p_0 = \frac{\left[ \sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} x_i y_i \right]}{\left[ n \sum_{i=1}^{n} x_i^2 - \left( \sum_{i=1}^{n} x_i \right)^2 \right]}$$

Result is known as linear regression of y on x

# Method of least squares

## Comments

Answer is analytic for any polynomial (no iteration involved)

(Aside: Iterative methods are genuinely numerical

Guess initial values for $p_1$ and $p_0$

Make $p_1$ larger say. Does S increase? If so, make $p_1$ smaller.

Found best value of $p_1$? Now vary $p_0$?

Found best value of $p_0$? Now iterate by varying $p_1$ again.

Stop when no further change.)

Extension to higher orders of polynomial is rather obvious

But, simultaneous equation involving sums are cumbersome

There must be a better way!

There is, but first introduce a slightly more convenient notation

Let $\left[x^m\right]=\sum_{i=1}^{n} x_i^m$ (Gaussian notation); hence $\left[x^0\right]=\sum_{i=1}^{n} x_i^0 = n$ etc

# Gaussian notation

## Rewrite the simultaneous equations

Now we have the following:

$$p_1[x^2] + p_0[x^1] = [x^1 y]$$

$$p_1[x^1] + p_0[x^0] = [x^0 y]$$

Introduced for symmetry

Now in matrix form:

Matrix A

$$\begin{pmatrix} [x^2] & [x^1] \\ [x^1] & [x^0] \end{pmatrix} \begin{pmatrix} p_1 \\ p_0 \end{pmatrix} = \begin{pmatrix} [x^1 y] \\ [x^0 y] \end{pmatrix}$$

Solving for $p_1$ and $p_0$ involves (simply) inverting the matrix A

$$\begin{pmatrix} p_1 \\ p_0 \end{pmatrix} = A^{-1} \begin{pmatrix} [x^1 y] \\ [x^0 y] \end{pmatrix}$$

# Gaussian notation

## Extension to higher order polynomials

The A matrix has a simple structure that is easy to extend:

First order polynomial (linear)

$$A = \begin{pmatrix} [x^2] & [x^1] \\ [x^1] & [\ ] \end{pmatrix}$$

Second order p… (quadratic)

$$A = \begin{pmatrix} [x^4] & [x^3] & [x^2] \\ [x^3] & [x^2] & [x^1] \\ [x^2] & [x^1] & [x^0] \end{pmatrix}$$

Later we'll see that $A^{-1}$ is in fact the error (or covariance) matrix

Matrix functions in EXCEL include =MINVERSE() and =MMULT()

Select enough destination cells (e.g. 2x2) and type:

CTRL+SHIFT+ENTER to complete (⌘+RETURN on a Mac)

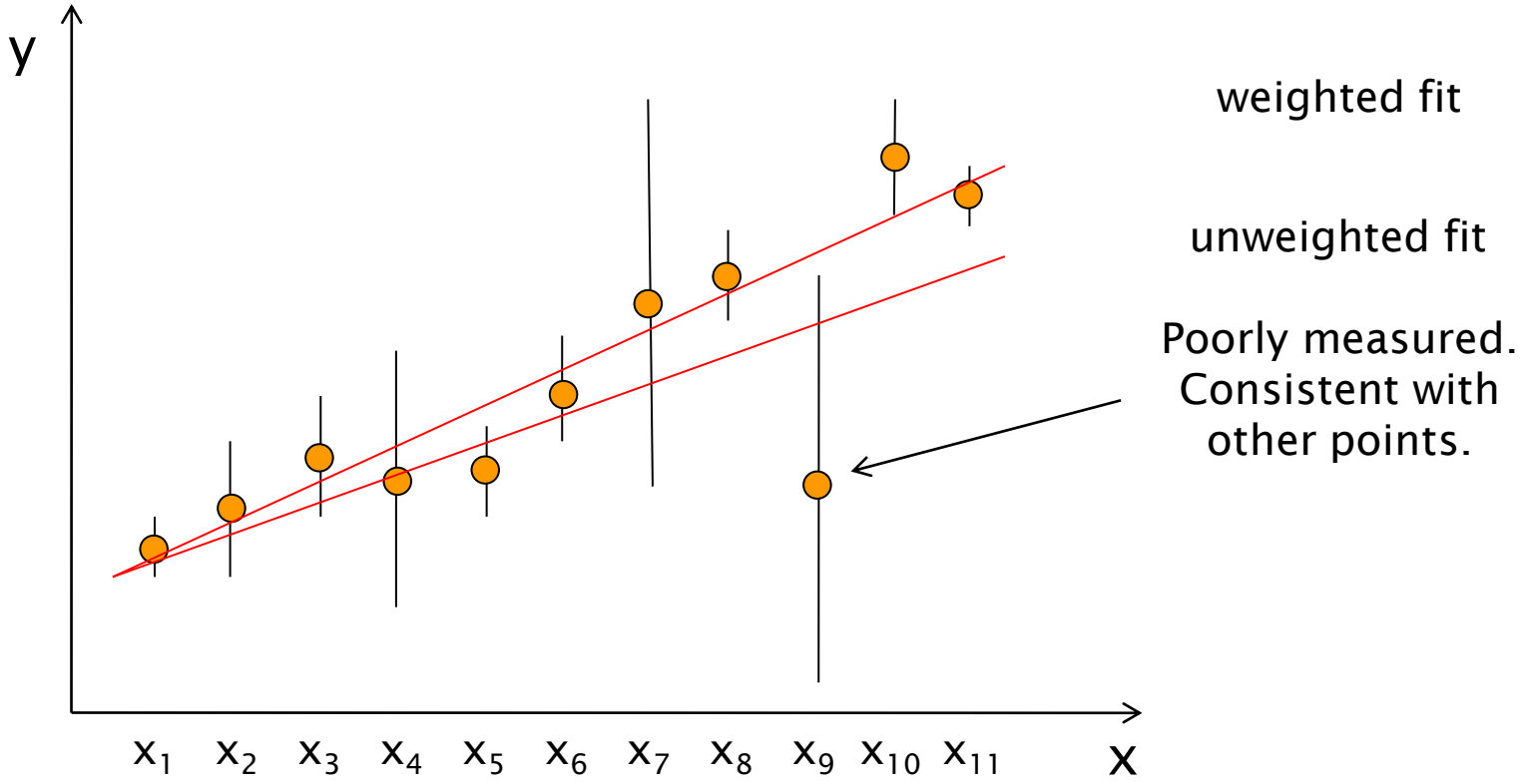
# Session B – Weighted least squares

## What is a weighted least squares fit?

A procedure for dealing with variable experiment errors:

Reflects that some data points are more reliable than others

Data points with small errors should have greater weight when determining the best fit parameters ($p_0$, $p_1$, … etc)



# Defining the weight

## Use the measurement error

Poorly measured (inaccurate) data points should have less weight

Scale the deviation by the measurement error

The weight sum of the squares of the deviations becomes:

$$S = \sum_{i=1}^{n} \left[ \frac{y_i - y_{fit}(x_i)}{\sigma_i} \right]^2$$

[ ] = no. of standard deviations

Note: if some data points are many standard deviations away

1. Either the error or the data point (or both) are wrong
2. Or the underlying theoretical assumptions (giving $y_{fit}$) wrong

In matrix form:

$$\begin{pmatrix} [\omega x^2] & [\omega x^1] \\ [\omega x^1] & [\omega x^0] \end{pmatrix} \begin{pmatrix} p_1 \\ p_0 \end{pmatrix} = \begin{pmatrix} [\omega x^1 y] \\ [\omega x^0 y] \end{pmatrix}$$

$$[\omega x^m] = \sum_{i=1}^{n} \omega_i x_i^m$$

$$\omega_i = \frac{1}{\sigma_i^2}$$

# Chi-squared

## Defining what is meant by a 'good' fit

Now we have defined a weighted least squares fit,

we can identify $S$ with the statistical quantity $\chi^2$ (chi-squared).

A 'good' fit will have $\chi^2/\text{dof} \approx 1$

dof = the number of degrees of freedom

dof = number of data points – number of fit parameters

A linear fit has 2 free parameters ($p_1$ and $p_0$)

The more parameters in the fit, the more data points you need

An unweighted least squares fit assumes all errors $\sigma_i = \sigma_j = 1$.

## What if $\chi^2$/dof is very different from unity?

You should think about what you are doing:

1. Either the you have over- (<1) or under- (>1) estimated your errors
2. Or, the function $y_{fit}$ doesn't represent the data (different physics)

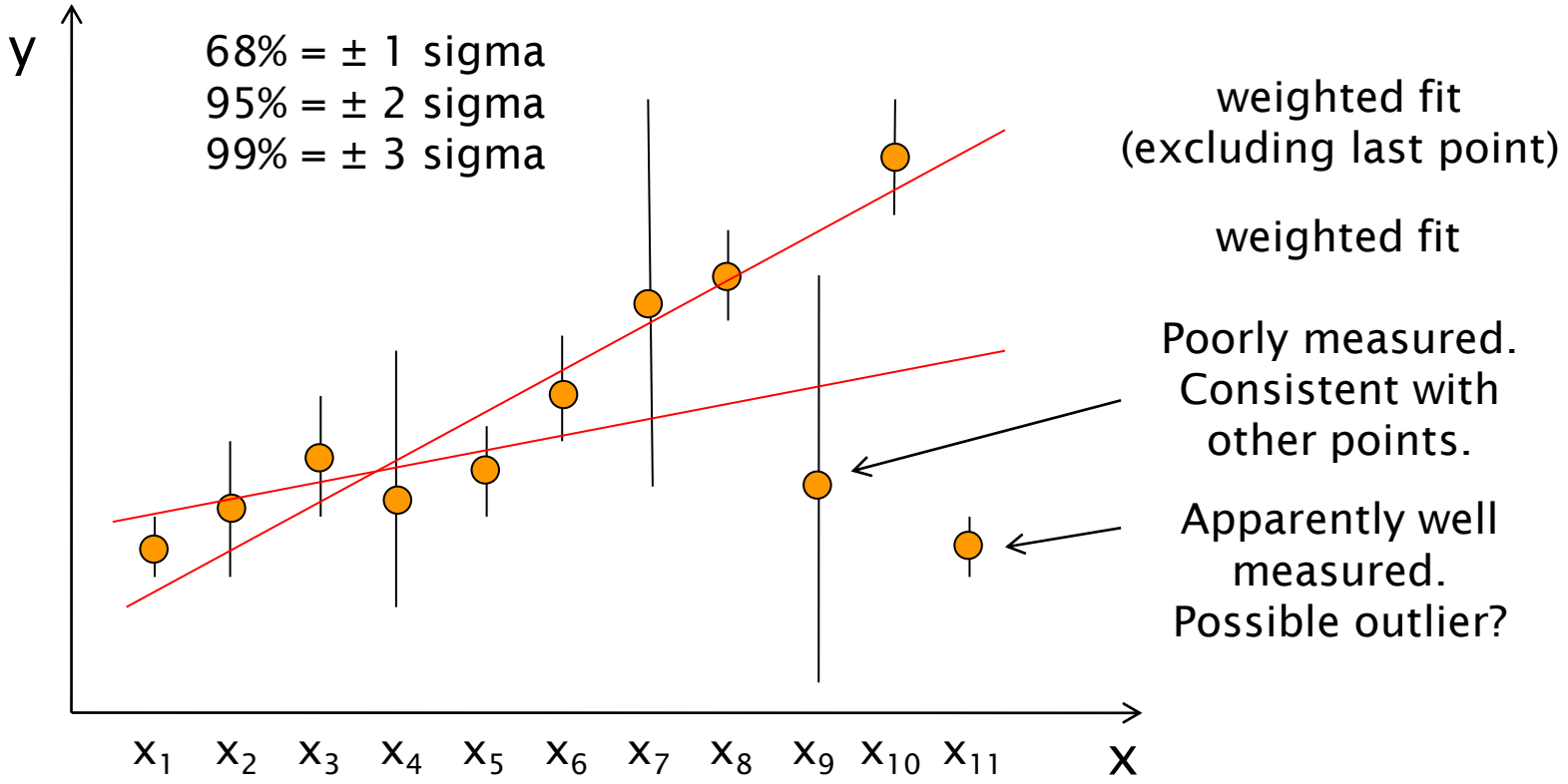
# Outliers

Unexpected data points that don't fit the trend

What to do about rogue data points?

If all points measured with the same equipment,

there is no real reason to discard the data point (re-measure).



# Session C - Covariance

## What to do when errors are correlated

What do I mean by correlated errors?

Usual (not always correct) assumption: errors are uncorrelated

This means they can be treated independently

Independent errors can be added in quadrature

Let y be a function of two variables a and b: e.g. $y = ax + b$

$$\sigma_y^2 = \left(\frac{\partial y}{\partial a}\right)^2 \sigma_a^2 + \left(\frac{\partial y}{\partial b}\right)^2 \sigma_b^2$$

All standard error propagation formulae are derived from here

Gradients are a measure of how quickly $y$ varies with $a$ or $b$

However, a and b may be correlated by physics (e.g. $a = -b^2$)

Now $a$ and $b$ are not independent – take into account covariance

$$\sigma_y^2 = \left(\frac{\partial y}{\partial a}\right)^2 \sigma_a^2 + 2\left(\frac{\partial y}{\partial a}\right)\left(\frac{\partial y}{\partial b}\right)\mathrm{cov}(a,b) + \left(\frac{\partial y}{\partial b}\right)^2 \sigma_b^2$$

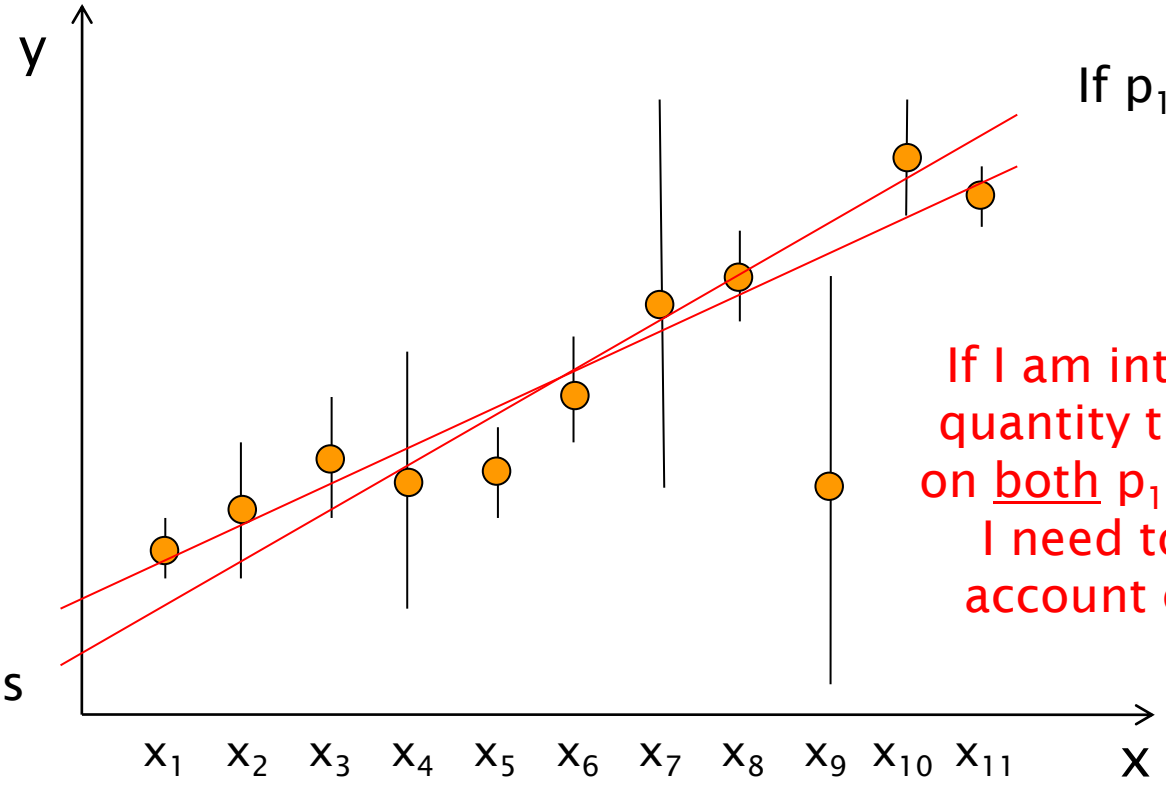
# Determining covariance

## How do I find the covariance term?

It depends on the problem:

We are mainly interested in errors due to fitting

For a straight line the error on $p_1$, $p_0$ are almost always correlated




If $p_1$ increases …

… $p_0$ decreases

If I am interested in a quantity that depends on <u>both</u> $p_1$ and $p_0$, then I need to take into account covariance.

# Getting used to covariance

First some examples other than fitting

Construct some simple problems where covariance is explicit

First example: calculate mean value of $y = x_1 + x_2$

based on a number of 'measurements' of $x_1$ and $x_2$

but where there is an explicit correlation $x_2 = 2x_1$ (really $y = 3x_1$)

The variance of a set of values is given by:

$$\sigma_{x_1}^2 = \sum_{i=1}^{n} \frac{(x_1(i) - \bar{x}_1)^2}{n(n-1)} \quad \sigma_{x_2}^2 = \sum_{i=1}^{n} \frac{(x_2(i) - \bar{x}_2)^2}{n(n-1)}$$

And the covariance is given by:

$$\mathrm{cov}(x_1, x_2) = \sum_{i=1}^{n} \frac{(x_1(i) - \bar{x}_1)(x_2(i) - \bar{x}_2)}{n(n-1)}$$

Note: the error on y is wrong if we treat $x_1$ and $x_2$ as independent

# Covariance matrix

## Defining the covariance matrix

Recall that the error propagation formula:

$$\sigma_y^2 = \left(\frac{\partial y}{\partial a}\right)^2 \sigma_a^2 + 2\left(\frac{\partial y}{\partial a}\right)\left(\frac{\partial y}{\partial b}\right)\mathrm{cov}(a,b) + \left(\frac{\partial y}{\partial b}\right)^2 \sigma_b^2$$

We can write this more succinctly in matrix form:

$$V_y = D^T V_{a,b} D$$

Where $V_{a,b}$ is the covariance matrix:

$$V_{a,b} = \begin{pmatrix} \sigma_a^2 & \mathrm{cov}(a,b) \\ \mathrm{cov}(a,b) & \sigma_b^2 \end{pmatrix} \qquad D = \begin{pmatrix} \frac{\partial y}{\partial a} \\ \frac{\partial y}{\partial b} \end{pmatrix}$$

And D is a column vector and $D_T$ its transpose.

In this case $V_y = \sigma_y^2$. In principle $V_y$ could be a matrix.

For two functions $y_1$ and $y_2$, D would be a 2x2 matrix … etc …

# Covariance and least squares fitting

## Identify the covariance matrix

Recall that we can solve for parameters using:

$$\begin{pmatrix} p_1 \\ p_0 \end{pmatrix} = A^{-1} \begin{pmatrix} [\omega x^1 y] \\ [\omega x^0 y] \end{pmatrix}$$

Assuming reliable errors; the covariance matrix is simply:

$$V_{p_1,p_0} = \begin{pmatrix} \sigma_{p_1}^2 & \mathrm{cov}(p_1,p_0) \\ \mathrm{cov}(p_1,p_0) & \sigma_{p_0}^2 \end{pmatrix} = A^{-1}$$

Alternatively, some quote the following:

$$V_{p_1,p_0} = \frac{\chi^2}{dof} A^{-1}$$

Let's think about what this means.

# Comment on errors

## If the errors are reliable

If the errors are well known, then we'd expect a 'good' fit to yield $\chi^2/dof \approx 1$, in which case:

$$V_{p_1,p_0} = \frac{\chi^2}{dof} A^{-1} \approx A^{-1}$$

## If the errors are not reliable

There are two possibilities:

1. Errors underestimated: $\chi^2/dof > 1$
2. Errors overestimated: $\chi^2/dof < 1$

Result is to scale the covariance matrix.

This has got to be good, right?

Not necessarily, we should also be open to the possibility that our function, $y_{fit}$, is simply the wrong one.

# Other forms for the covariance matrix

## What if I don't know *A*?

It could be that we use some iterative method to determine the parameters, but then we don't have the matrix *A* to find the covariance matrix.

The matrix *A* can also be written as:

$$A = \begin{pmatrix} \frac{1}{2}\frac{\partial^2 S}{\partial a^2} & \frac{1}{2}\frac{\partial^2 S}{\partial a \partial b} \\ \frac{1}{2}\frac{\partial^2 S}{\partial b \partial a} & \frac{1}{2}\frac{\partial^2 S}{\partial b^2} \end{pmatrix}$$

If we can determine the second derivatives of *S*, either analytically or numerically (see later!), we can then determine *A* and hence get the covariance matrix.

# Session D – Finite Differences

## Introduction to numerical analysis

Numerical method are needed when:

1. There is no analytic solution to a problem
2. You have a discrete set of measurements representing f(x)

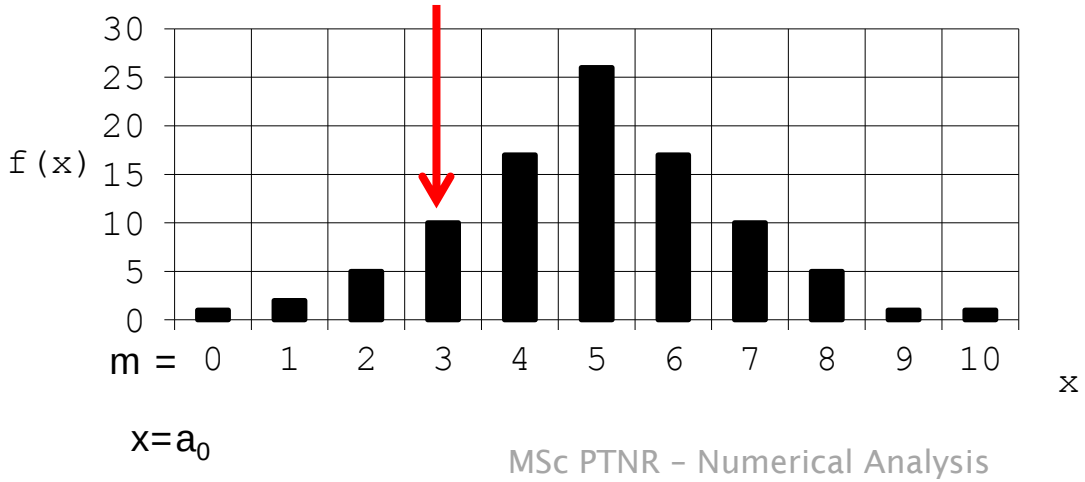
Question is how to extract information in this situation

Consider a discrete distribution

We have measurements at specific *mesh* points labelled $m = 0, 1, 2, \ldots$

Let the mesh points be equally spaced with step size = $h$

*How to calculate gradient here say?*




$f(x) = f(a_0+mh)$

$f(x) = f(m)$

$\text{grad} \approx [f(4)-f(3)]/h$

*Example of a forward difference operation*

# Finite difference operators

## Accessing and manipulating the data

We can define a number of operators that allow us to specify individual points (indexed by m) and do operations on them

| | | | |
|---|---|---|---|
| 1. | Forward shift | $E$ | $E\{f(m)\} = f(m+1)$ |
| 2. | Forward difference | $\Delta$ | $\Delta\{f(m)\} = f(m+1) - f(m)$ |
| 3. | Backward difference | $\nabla$ | $\nabla\{f(m)\} = f(m) - f(m-1)$ |
| 4. | Central difference | $\delta$ | $\delta\{f(m)\} = f(m+\frac{1}{2}) - f(m-\frac{1}{2})$ |
| 5. | Averaging | $\mu$ | $\mu\{f(m)\} = \frac{1}{2}[f(m+\frac{1}{2}) + f(m-\frac{1}{2})]$ |
| 6. | Differential | D | $D\{f(m)\} = (df/dx)_m$ |

Let's see how this works in a spreadsheet … (using $a_0=2$, $h=0.5$)

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | **m** | **$x=a_0+mh$** | **$f(x)=x^2$** | **E{f(m)}** | **FD** | **BD** |
| 2 | 0 | 2 | 4 | =C3 | =C3–C2 | |
| 3 | 1 | 2.5 | 6.25 | =C4 | =C4–C3 | =C3–C2 |
| 4 | 2 | 3 | 9 | =C5 | =C5–C4 | =C4–C3 |

# Finite difference operators

## Inverse operators

All the above operators commute:

e.g. $AB - BA = 0$ and $AA^{-1} = 1$

Consider the forward shift operator:

$E^{-1}E = 1$ note: 1 is a multiplicative operator

$E^{-1}\{E\{f(m)\}\} = f(m)$ but $E\{f(m)\} = f(m+1)$

$E^{-1}\{f(m+1)\} = f(m)$ $E^{-1}$ therefore does a backward shift

## Central differences

Note: central difference and averaging operators refer to points *between* our measurements. They are usually applied twice.

$\delta^2\{f(m)\} = \delta\{\delta\{f(m)\}\} = \delta\{f(m+½) - f(m-½)\}$

$\delta^2\{f(m)\} = [f(m+1) - f(m)] - [f(m) - f(m-1)]$

$\delta^2\{f(m)\} = f(m+1) - 2f(m) + f(m-1)$ Note: back on mesh again

# Session E – Interpolation

## Introduction

We met the forward shift operator in the last session

$f(1) = E\{f(0)\}$

$f(2) = E\{f(1)\} = E^2\{f(0)\}$

…

$f(n) = E^n\{f(0)\}$

What if we want to estimate an intermediate point?

Consider what happens when $n=m$ is a non-integer

We'll explore this using the relationship $E = 1 + \Delta$

$f(m) = (1+\Delta)^m\{f(0)\}$

Numerical methods make extensive use of expansions

$f(m) = (1 + m\,\Delta/1! + m\,(m\text{-}1)\Delta^2/2! + m\,(m\text{-}1)\,(m\text{-}2)\,\Delta^3/3! + \ldots)\{f(0)\}$

Reminder: check the Taylor series expansion of $(1+x)^n$

# Newton's interpolation formula

## First order approximation

Keeping only the first expansion term we find

$f(m) = (1 + m\,\Delta/1!)\{f(0)\}$

$f(m) = (1 + m\,\Delta)\{f(0)\}$

$f(m) = f(0) + m\,[f(1) - f(0)]$

Which can be written as (collecting the terms)

$f(m) = (1-m)\,f(0) + m\,f(1)$ $\qquad$ $f(m) = a_0 + mh$

You should recognise this as linear interpolation

In general, for $dx$ between $x=a_0$ and $x=a_0+h$, this becomes

$f(x+dx) = (1-dx/h)\,f(x=a_0) + dx/h\,f(x=a_0+h)$

# Newton's interpolation formula

## Second order approximation

Keep two terms from Newton's interpolation formula

$$f(m) \approx (1 + (m\,\Delta/1!) + (m\,(m-1)\,\Delta^2/2!))\{f(0)\}$$

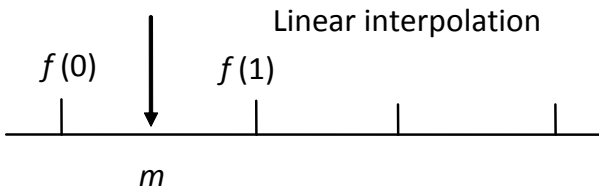
$$f(m) = f(0) + m\,[f(1)-f(0)] + (m\,(m-1)/2))\,\Delta\{f(1) - f(0)\}$$

$$f(m) = f(0) + m\,[f(1)-f(0)] + (m\,(m-1)/2))\,[f(2)-f(1)-f(1)+f(0)]$$

$$f(m) = f(0)\,[1-m+(m\,(m-1)/2))] + f(1)\,[m-m\,(m-1)] + f(2)\,[m\,(m-1)/2)]$$

This is parabolic interpolation

- Uses 3 points instead of 2

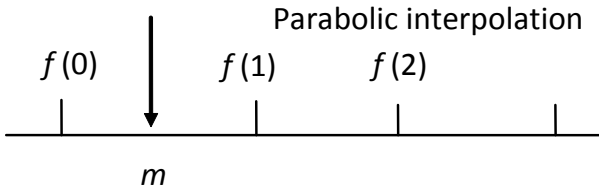


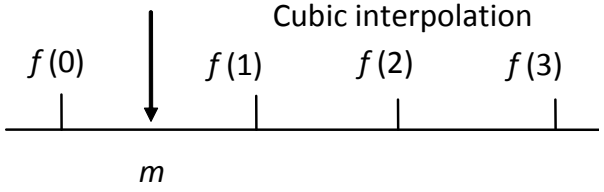

## Comments

Increase accuracy by keeping more terms

This particular formulation uses $\Delta$

- Only uses *forward* values of $f(m)$



# Newton's interpolation formula

## Using other difference operators

We have a lot of choice in which operator(s) we use

An alternative choice would be $E = 1 + \nabla$

This would use points backwards of the first point (at m=0)

We can combine both approaches using $E = (1 + \delta^2/2 + \delta\,\mu)$

Uses combination of central difference and averaging operators

This is the origin of Stirling's interpolation formula

$$f(m) = (1 + m^2\delta^2/2! + m^2(m^2-1)\delta^4/4! + \dots)\{f(0)\}$$
$$+ (m\,\delta/1! + m(m^2-1)\delta^3/3! + \dots)\,\mu\{f(0)\}$$

In general, for best accuracy, keep more terms

However, if the underlying physical function is quadratic then a second order interpolation will give an exact solution

The choice of operators depends on various factors such as simplicity and efficiency, as well as accuracy

## Differentiation by Finite Differences

How to differentiate an unknown function based on discrete values

We need a finite difference form of the differential operator

$D\{f(\mathrm{m})\} = (\mathrm{d}f/\mathrm{d}x)_{\mathrm{m}}$ evaluated at $x = a_0 + mh$

Consider the Taylor Series expansion

$$y(x+h) = y(x) + h(\mathrm{d}y/\mathrm{d}x)_{y(x)} + (h^2/2!)\,(\mathrm{d}^2y/\mathrm{d}x^2)_{y(x)} + \ldots + (h^{\mathrm{n}}/\mathrm{n}!)\,(\mathrm{d}^{\mathrm{n}}y/\mathrm{d}x^{\mathrm{n}})_{y(x)}$$

Write in operator form

$$y(x+h) = (1 + hD + (h^2/2!)D^2 + \ldots + (h^{\mathrm{n}}/\mathrm{n}!)\,D^{\mathrm{n}})\,\{y(\mathrm{x})\} \qquad (1)$$

Now note that

$y(x+h) = E\{y(x)\}$ is just a forward shift (2)

$e^Z = (1 + z + (z^2/2!) + \ldots + (z^{\mathrm{n}}/\mathrm{n}!) + \ldots)$ is a standard expansion (3)

Combine (1), (2) and (3)

$$E\{y(x)\} = (e^{hD})\{y(x)\}$$

# Differentiation by finite differences

## The differential operator

Now we can identify $D$ in term of $E$, which we can handle

$E = e^{hD}$

$hD = \ln(E)$

$hD = \ln(1+\Delta)$ in terms of the forward difference operator

$hD = -\ln(1-\nabla)$ in terms of the backward difference operator

Nearly there!

But, how do we perform the natural logarithm of an operator?

We need another expansion. Recall the Maclaurin series …

$\ln(1+x) = x - x^2/2 + x^3/3 - x^4/4 + \ldots$

$hD = \Delta - \Delta^2/2 + \Delta^3/3 - \Delta^4/4 + \ldots$

Unsurprisingly, the first order approximation is

$(df/dx)_{m=0} = D\{f(0)\} \approx (1/h)[f(1) - f(0)]$

# Differentiation by finite differences

## Higher order differentials

How do we extend this to 2nd and 3rd order differentials?

$D^r\{f(m)\} = (d^r f/dx^r)_m$

$h^r D^r\{f(0)\} = (\Delta - \Delta^2/2 + \Delta^3/3 - \Delta^4/4 + \ldots)^r\{f(0)\}$

Take common (operator) factor of $\Delta^r$ outside

Expand based on $(1+z)^r$ where $z = [-(1/2)\Delta+(1/3)\Delta^2-(1/4)\Delta^3+\ldots]$

$h^r D^r\{f(0)\} = h^r (d^r y/dx^r)_{m=0} = \Delta^r(1 + rz + (r(r-1)/2!)z^2 + \ldots)\{f(0)\}$

Collect terms to the same order in $\Delta$

$h^r (d^r y/dx^r)_{m=0} = (\Delta^r - (r/2)\Delta^{r+1} + (r(3r+5)/24)\Delta^{r+2} \ldots)\{f(0)\}$

Note 1: numerical analysis makes extensive use of expansions!

Note 2: accuracy depends on the number of terms kept

Note 3: there are other formulations using different operators

Note 4: accuracy also depends on the choice of operators

# Partial differentiation

## Functions of more than one variable

How do we deal with partial derivatives?

$$\frac{\partial f(x,y)}{\partial x};\frac{\partial^2 f(x,y)}{\partial y^2};\frac{\partial^3 f(x,y)}{\partial x \partial y^2};\ldots$$

As before, we can define differential operators:

$D_x\{f(m,n)\} = (\partial f/\partial x)_{m,n}$

$D_y\{f(m,n)\} = (\partial f/\partial y)_{m,n}$

evaluated at

$x = a_0 + mh_x$

$y = b_0 + nh_y$

These only affect the variable they refer to, so for the first order approximation (with forward differences):

$$D_x\{f(m,n)\} = (\partial f/\partial x)_{m,n} \approx (1/h_x)[f(m+1,n) - f(m, n)]$$

$$D_y\{f(m,n)\} = (\partial f/\partial y)_{m,n} \approx (1/h_y)[f(m,n+1) - f(m, n)]$$

$$D_xD_y\{f(m,n)\} = (\partial^2 f/\partial x \partial y)_{m,n} \approx (1/h_xh_y)[f(m+1,n+1) - f(m, n+1) - f(m+1, n) + f(m,n)]$$

(now you can work out the covariance matrix…)

## Numerical Integration

Some aspects of this exercise should be familiar to all of you

We will consider 3 methods:

Trapezium method, Simpson's rule, Gaussian quadrature

## Trapezium method

This method is simple and fast, but not particularly accurate

Calculate area of trapezium beneath neighbouring mesh points

(a) For one panel

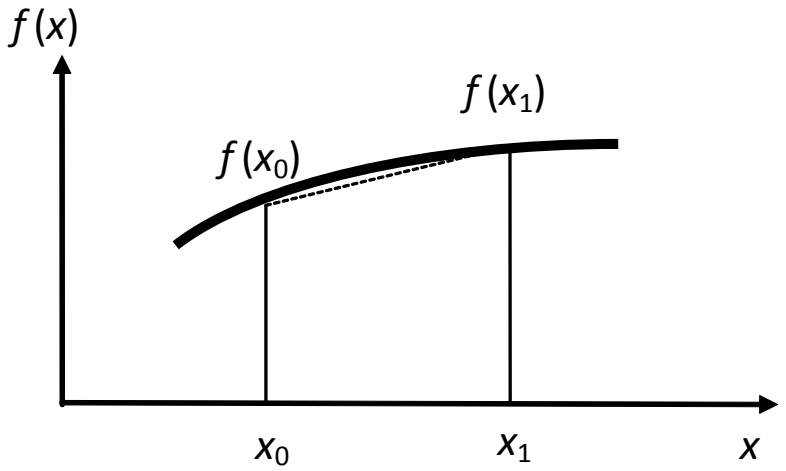
$$I = \int_{x_0}^{x_1} f(x)dx \approx 0.5\left[f(x_0)+f(x_1)\right]\left[x_1 - x_0\right]$$

(b) For several panels

$$I = \int_a^b f(x)dx \approx \sum_{i=0}^{i=n-1} 0.5\left[f(x_i)+f(x_{i+1})\right]\left[x_{i+1} - x_i\right]$$

(c) With constant panel width

$$I = \int_a^b f(x)dx \approx \sum_{i=0}^{i=n-1} 0.5\left[f(x_i)+f(x_{i+1})\right]\ h$$

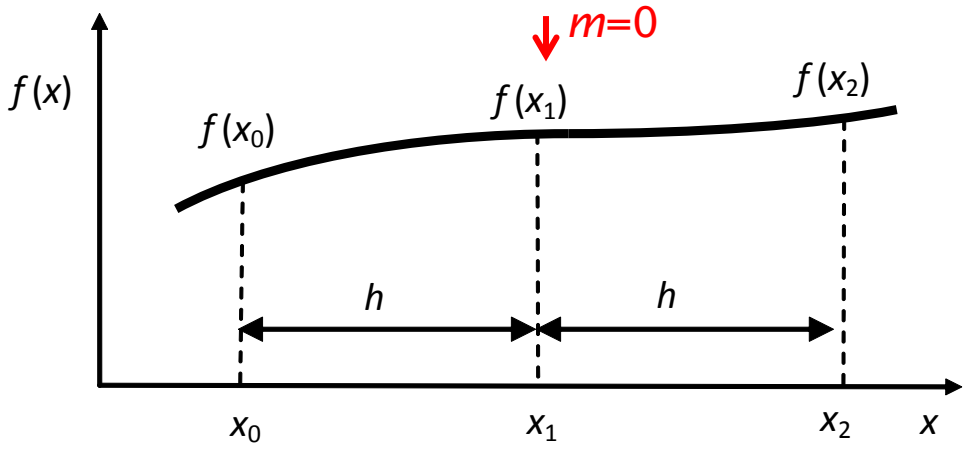


# Numerical Integration

## Simpson's rule

Also simple, but more accurate unless function is a straight line

Equivalent to fitting a parabola through three adjacent points




Based on second order central differences

$$f(a_0+mh) = f(a_0) + \delta\{f(a_0)\}m + \delta^2\{f(a_0)\}m^2/2! \qquad m = \pm 1$$

where

$$\delta\{f(a_0)\} = [f(a_0 + h/2) - f(a_0 - h/2)]$$

$$\delta^2\{f(a_0)\} = [f(a_0+h) - 2f(a_0) + f(a_0-h)]$$

# Simpson's Rule

Over two panels (three mesh points) we want the integral

$$I = \int_{m=-1}^{m=1} f(a_0 + mh)h\,dm$$

Substitute for $f(a_0+mh)$ and integrate

$$I = h[2f(a_0) + (1/3)\delta^2\{f(a_0)\}]$$

$\delta$ term cancels

$$I = h[(1/3)f(a_0 - h) + (4/3)f(a_0) + (1/3)f(a_0 + h)]$$

$$I = h[(1/3)f(x_0) + (4/3)f(x_1) + (1/3)f(x_2)]$$

$m$=−1 $m$=0 $m$=+1

Now generalise for a succession of panels

$$I = (h/3)\ [f(x_0) + 4\ \Sigma_{\text{odd index terms}} + 2\ \Sigma_{\text{even index terms}} + f(x_{2n})]$$

$m$=0 $m$=1,3,5,… $m$=2,4,6,… $m$=2$n$

Must have an EVEN number of panels

If you have an odd number of panels, …

… first or last panel must use trapezium method

# Gaussian Quadrature

## A different approach

Potentially the most accurate method of evaluating integrals

The basic idea is as follows:

- Give up on the idea of fixed step sizes ($h \neq$ constant)
- Optimise the position where function will be sampled
- (Note: applies to analytic functions not discrete sets of points)
- (Note: integral doesn't have to be analytic)

For generality, express your integral over the range $-1 < x < 1$

$$I = \int_{-1}^{1} f(x)dx \approx \sum_{k=1}^{n} f(x_k)\omega_k$$

- Requires a change of variable (not a big deal)
- Note that the $x_k$ values are not evenly spaced
- Each function value, $f(x_k)$, is weighted by a different factor, $\omega_k$

# Gaussian Quadrature

## Calculating the sampling points and their weights

If we have $n$ points, we have $2n$ parameters to be determined

These are the $x_k$ and $\omega_k$ values; $k = 1,\ldots,n$

Integral range is symmetric by construction

Sampling points and weights are the same for $\pm x$

Consider $n$ = odd and $n$ = even

$$I = \int_{-1}^{1} f(x)dx \approx f(0)\omega_0 + \sum_{k=1}^{m}\left[f(x_k) + f(-x_k)\right]\omega_k \qquad I = \int_{-1}^{1} f(x)dx \approx \sum_{k=1}^{m}\left[f(x_k) + f(-x_k)\right]\,\omega_k$$

ODD
$m = (n - 1)/2$

EVEN
$m = n/2$

For two points, we now have 4 parameters

Integral should be exact for a third order polynomial; use this to …

Find the weights by working out the integral for $f(x) = 1,\ x,\ x^2,\ x^3$ …

For which we know the answer!

# Gaussian Quadrature

## Calculating the sampling points and their weights

Here is the result for two sampling points, $n = 2$

$$\int_{-1}^{1} 1\,dx = 2 = \omega_1[1+1] \qquad \Rightarrow \omega_1 = 1$$

$$\int_{-1}^{1} x\,dx = 0 = \omega_1[x_1 - x_1]$$

$$\int_{-1}^{1} x^2\,dx = \frac{2}{3} = \omega_1\left[x_1^{\,2} + x_1^{\,2}\right] \qquad \Rightarrow x_1 = 1/\sqrt{3}$$

$$\int_{-1}^{1} x^3\,dx = 0 = \omega_1\left[x_1^{\,3} - x_1^{\,3}\right]$$

Sufficient to determine integral of any polynomial of order <= 3

Note: we could have chosen more complicated functions, but …

… the result would have been the same

The process can be repeated for $n = 3$ and higher

# Gaussian Quadrature

## Gauss-Legendre

The method we have used is the Gauss-Legendre method

The remarkable result is that:

- The sampling points and their weights are fixed for any given *n*
- They do not depend on the function we wish to integrate

They are tabulated in reference books (e.g. see below)

| $n = 2$ | $x_i$ | $\omega_i$ |
|---|---|---|
| | ±0.577350269189626 | 1.000000000000000 |

| $n = 3$ | $x_i$ | $\omega_i$ |
|---|---|---|
| | 0.000000000000000 | 0.888888888888889 |
| | ±0.774596669241483 | 0.555555555555556 |

| $n = 4$ | $x_i$ | $\omega_i$ |
|---|---|---|
| | ±0.339981043584856 | 0.652145154862546 |
| | ±0.861136311594053 | 0.347854845137454 |

Remember: you need to convert your integral range into $-1 < x < 1$

## Finite difference solutions of differential equations

Suppose we want to solve a homogeneous equation of the form

$$\frac{d^2y}{dx^2} - \frac{dy}{dx} = 0$$

By solve, we mean find $y(x)$ for all $x$

Our finite difference solution will evaluate $y(x)$ at discrete points

## Order of the difference equation

Depends on the choice of operator(s)

Not necessarily the same as the order of the differential equation

$$\frac{1}{h^2}\Delta^2 y(x) - \frac{1}{h}\nabla y(x) = 0$$

At a specific point $x = a_0 + jh$, we get a difference equation

$$y_{j+2} - 2y_{j+1} + (1-h)y_j + hy_{j-1} = 0$$

# Order of the difference equation

Now set $h = 1$; amounts to a change of variable $x$ -> $x$'

$$y_{j+2} - 2y_{j+1} + y_{j-1} = 0$$

Result is 3rd order difference equation (for this choice of operators)

## First order differential equations (separable)

Most straightforward class of equations to solve

Consider a general first order differential equation of the form

$$\frac{dy}{dx} = f(x)$$

Can be solved analytically (so we know what to expect)

$$y(x) - y(0) = \int_0^x f(x)dx$$

$$y(x) = y(0) + \int_0^x f(x)dx$$

Solution requires that we know an initial value $y(0)$

# First order equations (separable)

## Finite difference solution

The finite difference form of the differential equation is

$$\left.\frac{dy}{dx}\right|_{x_j} \approx \frac{(y_{j+1} - y_j)}{h} \equiv f_j$$

It is straightforward to see that the solution is just

$$y_1 = y_0 + hf_0$$
$$y_2 = y_1 + hf_1 = y_0 + h(f_1 + f_0)$$
$$\ldots etc \ldots$$
$$y_n = y_0 + h\sum_{k=0}^{n-1} f_k$$

The final line is the solution we require

# First order equations (separable)

## General first order differential equations

Consider the general homogeneous equation below

$$\frac{dy}{dx} - a(x)y(x) = 0$$

This equation is also separable. The analytic solution is

$$y(x) = y(0)\exp\left(\int_0^x a(x)dx\right)$$

The finite difference form of the differential equation is

$$\frac{(y_{j+1} - y_j)}{h} \approx a_j y_j$$

The solution is

$$y_1 = y_0 + ha_0 y_0 = y_0(1 + ha_0)$$
$$y_2 = y_1 + ha_1 y_1 = y_0(1 + ha_0)(1 + ha_1)$$
$$\ldots etc \ldots$$
$$y_n = y_0 \prod_{k=0}^{n-1} (1 + ha_k)$$

# General first order differential equations

This doesn't look quite like the analytic solution, but …

… now rewrite the solution in the following way

$$y_n = y_0 \prod_{k=0}^{n-1} (1 + h a_k)$$

$$y_n = y_0 \exp\left( \ln\left( \prod_{k=0}^{n-1} (1 + h a_k) \right) \right) \qquad \ln(a.b) = \ln(a) + \ln(b)$$

$$y_n = y_0 \exp\left( \sum_{k=0}^{n-1} \ln(1 + h a_k) \right)$$

This is directly comparable with the analytic solution (below)

$$y(x) = y(0) \exp\left( \int_0^x a(x) dx \right)$$

# General homogeneous equations

## Second order linear differential equations

Consider the following equation

$$\frac{d^2y}{dx^2} - 2\frac{dy}{dx} - y = 0$$

In finite difference form this becomes

$$\frac{1}{h^2}\Delta^2 y_j - 2\frac{1}{h}\Delta y_j - y_j = 0$$

$$y_{j+2} - 2(1+h)y_{j+1} + (1+2h-h^2)y_j = 0$$

$$y_{j+2} - 4y_{j+1} + 2y_j = 0$$

We can generalise this solution as

$$\sum_{i=0}^{n} a_i y_{j+n-i} = 0$$

n = 2 is order of the difference equation

Solution of the original equation will be of form $y = \beta^j$, or $y = e^{mj}$

# General homogeneous equations

Substitute $y_j = \beta^j$ into the finite difference form of the equation

$$\sum_{i=0}^{n} a_i y_{j+n-i} = 0$$

$$\sum_{i=0}^{n} a_i \beta^{j+n-i} = 0$$

Only get non-trivial solutions if $y_j = \beta^j \neq 0$. This means that

$$\sum_{i=0}^{n} a_i \beta^{n-i} = 0$$

taking out common factor of $\beta^j$

So, we now have an $n^{\text{th}}$ order polynomial to solve (easy!)

There will be $n$ of these solutions, giving the general solution

$$y_j = \sum_{k=0}^{n} b_k \beta_k^j$$

The general solution is a linear combination of these solutions

The coefficients, $b_k$, will depend on specific boundary conditions

# Session K – Runge Kutta Method

## Initial value problems

When you know the value and derivative of a function at a point

$$y' = \mathrm{d}y/\mathrm{d}x = f(x,y); \qquad y = y(x)$$

The simplest way to advance the solution is Euler's approximation

$$y_{n+1}(x_0+h) \approx y_n(x_0) + hy'_n(x_0)$$

You should recognise this as first two terms of a Taylor expansion

Higher accuracy requires higher order derivatives

But, there is another approach involving only first order derivatives

## Runge Kutta

Idea is to evaluate the derivative at (several) intermediate points

We then need to decide how to weight these intermediate results

WARNING: this looks terribly complicated, but it simplifies nicely

# Runge Kutta Method

## Advancing the solution

Instead of a Taylor series expansion we can write:

$$y_{n+1} \approx y(x_n) + h\begin{bmatrix} \alpha_0 f(x_n, y_n) \\ +\alpha_1 f(x_n + \mu_1 h, y_n + \lambda_1 h) \\ +\alpha_2 f(x_n + \mu_2 h, y_n + \lambda_2 h) \\ +\cdots \\ +\alpha_p f(x_n + \mu_p h, y_n + \lambda_p h) \end{bmatrix}$$

$p = 1$

Need to evaluate the gradient at $x_n$ and at $p$ intermediate points

- where $\lambda_n$ is a fraction of a step in $y$ direction
- and $\mu_n$ is fraction of a step in $x$ direction
- and $\alpha_n$ is the weight of each term

The simplest case (see worksheet) is when $p = 1$

# Runge Kutta Method

## Now it gets a bit complicated

With only one intermediate point the solution becomes

$$y_{n+1} = y_n + \alpha_0 k_0 + \alpha_1 k_1$$

where the $k_n$ terms are given by

$$k_0 = hf(x_n, y_n)$$

$$k_1 = h\left[ f(x_n, y_n) + (\mu h f_x + \lambda k_0 f_y) + \frac{1}{2}(\mu^2 h^2 f_{xx} + 2\mu\lambda h k_0 f_{xy} + \lambda^2 k_0{}^2 f_{yy}) + O(h^3) \right]$$

The $k_1$ term stems from a 2-d Taylor series expansion

$$f_x = \frac{\partial}{\partial x} f(x,y)\bigg|_{x_n, y_n} \qquad f_y = \frac{\partial}{\partial y} f(x,y)\bigg|_{x_n, y_n}$$

$$f_{xx} = \frac{\partial^2}{\partial x^2} f(x,y)\bigg|_{x_n, y_n} \qquad f_{xy} = f_{yx} = \frac{\partial^2}{\partial x \partial y} f(x,y)\bigg|_{x_n, y_n} \qquad f_{yy} = \frac{\partial^2}{\partial y^2} f(x,y)\bigg|_{x_n, y_n}$$

# Runge Kutta Method

## What about the higher order derivatives?

Now we can compare the RK method and Taylor series expansion

The standard Taylor series expansion is

$$y_{n+1} \approx y_n + hy'_n + \frac{h^2}{2!}y''_n + \frac{h^3}{3!}y'''_n \ldots$$

Rewriting in the same notation the TS expansion becomes

$$y_{n+1} = y_n + hf + \frac{h^2}{2}(f_x + ff_y)$$

$$+ \frac{h^3}{6}(f_{xx} + 2ff_{xy} + f^2 f_{yy} + f_x f_y + ff_y{}^2) + O(h^4)]$$

$y' = f(x,y)$ and $y = y(x)$

$$y'' = \frac{\partial f}{\partial x}\frac{\partial x}{\partial x} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial x} = f_x + ff_y$$

The Runge Kutta method gives

$$y_{n+1} = y_n + h(\alpha_0 + \alpha_1)f + h^2\alpha_1(\mu f_x + \lambda ff_y) +$$

$$\frac{h^3\alpha_1}{2}(\mu^2 f_{xx} + 2\mu\lambda ff_{xy} + \lambda^2 f^2 f_{yy}) + O(h^4)$$

# Runge Kutta Method

## Comparing the two …

What this shows is that there is only one arbitrary constant

Note: RK method is equivalent to TS to order $h^2$

But, the RK method only uses first derivatives

With one intermediate point we have a second order RK method

$$y_{n+1} = y_n + \frac{1}{2}(k_0 + k_1)$$

where

$$k_0 = hf(x_n, y_n)$$
$$k_1 = hf(x_n + h, y_n + k_0)$$

What does this mean?

We use the gradient at the initial point to calculate an approximate solution at the end point (using Euler's method). We then calculate the gradient at this approximate solution and use both to find the final solution. Hence, we are effectively using an approximation to the average gradient across the step.

# Runge Kutta Method

## Higher orders

We can get better accuracy by going to higher order

This means evaluating the gradient at more intermediate points

Most common applications stop at fourth order

$$y_1 = y_0 + \Delta y$$

where

$$\Delta y = \frac{1}{6}(k_0 + 2k_1 + 2k_2 + k_3)$$

and

$$k_0 = hf(x_0, y_0)$$

$$k_1 = hf\left(x_0 + \frac{h}{2}, y_0 + \frac{k_0}{2}\right)$$

$$k_2 = hf\left(x_0 + \frac{h}{2}, y_0 + \frac{k_1}{2}\right)$$

$$k_3 = hf(x_0 + h, y_0 + k_2)$$

Although somewhat complicated to derive the Runge Kutta method is rather straightforward to apply. Hence its popularity.

# Runge Kutta Method

## Systems of equations

The Runge Kutta method can also be used to solve systems of coupled first order differential equations:

$$\frac{dy}{dx} = f(x, y, z, \ldots) \qquad y_1 = y_0 + \Delta y \qquad \Delta y = \frac{1}{6}(k_0 + 2k_1 + 2k_2 + k_3)$$

$$\frac{dz}{dx} = g(x, y, z, \ldots) \qquad z_1 = z_0 + \Delta z \qquad \Delta z = \frac{1}{6}(q_0 + 2q_1 + 2q_2 + q_3)$$

$$\vdots$$

$$k_0 = hf(x_0, y_0, z_0, \ldots) \qquad q_0 = hg(x_0, y_0, z_0, \ldots)$$

$$k_1 = hf\left(x_0 + \frac{h}{2}, y_0 + \frac{k_0}{2}, z_0 + \frac{q_0}{2}, \ldots\right) \qquad q_1 = hg\left(x_0 + \frac{h}{2}, y_0 + \frac{k_0}{2}, z_0 + \frac{q_0}{2}, \ldots\right)$$

$$k_2 = hf\left(x_0 + \frac{h}{2}, y_0 + \frac{k_1}{2}, z_0 + \frac{q_1}{2}, \ldots\right) \qquad q_2 = hg\left(x_0 + \frac{h}{2}, y_0 + \frac{k_1}{2}, z_0 + \frac{q_1}{2}, \ldots\right)$$

$$k_3 = hf(x_0 + h, y_0 + k_2, z_0 + q_2, \ldots) \qquad q_3 = hg(x_0 + h, y_0 + k_2, z_0 + q_2, \ldots)$$